On Solving Nearly-Singular, Sparse Systems of Linear Equations: Diakoptics Techniques for Parallel Computing

L. Jonathan Dowell Douglas J. Roberts Dale B. Henderson

Los Alamos National Laboratory
P. O. Box 1663 MS F604
Los Alamos, NM 87545, USA

{ljdowell, dzzr, dbh}@lanl.gov

Abstract

The ELISIMS project of Los Alamos National Laboratory has applied diakoptics techniques for the solution of nearly-singular, sparse systems of linear equations on parallel-computing architectures. These techniques offer advantages over conventional Krylov solution techniques for the solution of nearly-singular linear systems. This paper discusses the diakoptics algorithm presented by Aitchison, describes the implementation of this algorithm on an extreme-Linux supercomputer, and discusses performance considerations.

1 Introduction

The solution of large, sparse systems of linear equations is important in many scientific applications. An example of such an application is the solution of the nodal and mesh equations describing Kirchoff's Laws that finds the voltages and currents of an electrical circuit. Also, the solution of large systems of non-linear equations by Newton's method reduces the problem to an iterative procedure involving successive solutions of systems of linear equations.

Methods for the solution of these large systems of linear equations exist, from Gauss-Jordan elimination [9] to Krylov methods [4]. However, considerations of numerical analysis, including accumulated round-off error and small differences between large numbers, can pose problems for practical implementations of these methods, especially for very large and nearly-singular systems.

Diakoptics is an alternative to these methods that offers several advantages, including scalability for parallel computing. Invented by Kron [7] and related to the inverse of the sum of two matrices [5], diakoptics solves a matrix equation by separating the matrix into a number of sub-matrices that are individually easier to solve, and composing their solutions to produce the solution of the original system. The practical applicability of the diakoptics method depends on the existence of a graph comprised of sparsely-interconnected sub-graph clusters, which is a characteristic of many physical systems for which a solution is sought. The method is even more efficient when the sub-graph clusters are themselves sparsely connected.

In this paper, we will review the diakoptics method as presented by Aitchison [1]. We will discuss a practical implementation of a diakoptics algorithm on an extreme-Linux supercomputer, with consideration of scaling performance and application to the computation of power flow in a commercial electric-power infrastructure.

2 Diakoptics

Aitchison [1] presented a succinct description of the diakoptics algorithm which we will repeat here for clarity. We wish to solve the equation $Mz = d$, where M is a non-singular $n \times n$ coefficient matrix, z is the vector of unknowns, and d is the vector of terms involving none of the unknowns. The matrix M can be decomposed as the sum of two matrices $M = A + B$. Then the solution z can be found in five steps as follows:

1. Solve for the vector x that is the solution of $Ax = d$.

2. Let B^r represent the r non-zero columns of B. Then solve the equation $AX = B^r$, where X is the $n \times r$ matrix whose columns are the solutions corresponding to each of the columns in B^r.

3. Select the rows from X whose row numbers match the column numbers of the non-zero columns of B. Call this $r \times r$ matrix X_r. Then solve the equation $(I + X_r)z_r = x_r$, where I is the $r \times r$ identity matrix and x_r are the elements of the solution vector x (that was found in step 1) whose indices match the column numbers of the non-zero columns of B. Note that z_r is the sub-vector of z comprised of the elements of z whose indices match the non-zero columns of B.

4. Calculate the n-r remaining elements of z using the formula $z_{n-r} = x_{n-r} - X_{n-r} z_r$. Here, the subscript n-r indicates those rows or columns corresponding to columns of B that were zero.

5. Finally, arrange the elements from z_{n-r} and z_r to achieve the solution vector z.

This algorithm will be advantageous if the matrix A has a convenient structure such as block diagonal. For a block-diagonal matrix consisting of k blocks, step 1 separates into k independent equations. Separate nodes in a parallel computer can solve each of these k solutions. Similarly, the factorization and storage required by step 1 can be re-used by each node to solve step 2 in parallel. Algorithmically, step 3 is a chokepoint for a master-slave parallel computing architecture, as it is a single step logically assigned to the master node that must be completed while the slave nodes wait. The partition of the matrix X among the slaves as a result of step 2 can be utilized in step 4 to solve that step in parallel. Step 5 is merely a storage convention, and can be accomplished by the slave nodes in parallel.

An elementary analysis of the effectiveness of the diakoptics method follows from assumptions about the solution algorithm for each step. The solution of a system of linear equations by a method such LU-decomposition and backsubstitution is $O(n^3)$ [9], while the subsequent solutions require repeating only the backsubstitution which is $O(n^2)$. Let us assume that the matrix A can be decomposed into k blocks of approximately the same size, so that each sub-block of the block diagonal matrix will have n/k rows. Then step 1 is $O[(n/k)^3]$, step 2 is $O[\, r \cdot (n/k)^2 \,]$, and step 3 is $O(r^3)$. If r is small, then the diakoptics method should be a $1/k^3$ improvement over solution by LU-decomposition. Of course, a practical implementation that takes advantage of the sparsity of A and its sub-blocks will modify this analysis.

3 Advantages of Diakoptics for Nearly-Singular, Sparse Systems

3.1 Example: Power-Flow Calculation

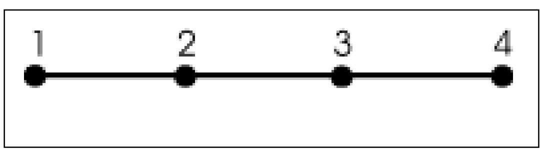


Figure 1. Simple Transmission Network.

As an example of an application of diakoptics to a nearly-singular, sparse linear system, consider the electric-power transmission system shown in Fig. 1. The techniques for calculating power flow are described in [13]. Power flows through transmission lines in proportion to the phase-angle differences between sinusoidal node voltages. The long transmission lines are primarily inductors, leading to this dependence of power flow upon phase angles. Figure 1 shows a 4-node transmission system containing 3 transmission lines that connect the nodes to form a simple tree. Let the reactance of each transmission line be normalized to 1.0. Let the normalized resistance of each transmission line be 0.01, which can be neglected. There are only two power injections, at the nodes at either end of the tree. The power injected at node 1 is 1 MW, and the power injected at node 4 is -1 MW. Let node 4 be the reference node, where the phase angle is equal to zero by definition. This problem qualifies for solution by linear approximation of the non-linear power-flow equations [13]. Then the power flow P_{ij} through a transmission line between nodes i and j is

$$P_{ij} = \frac{\theta_i - \theta_j}{x_{ij}}, \tag{1}$$

where θ_i and θ_j are the phase angles at nodes i and j, respectively, and x_{ij} is the normalized inductive reactance of the transmission line between nodes i and j.

The flow through each line must be 1 MW. Using a 100 MVA base for normalization, a phase-angle difference of 0.01 radians is required across each transmission line. As the phase angle is zero at the reference node, the consequent phase-angle vector, θ, is

$$\theta = \begin{pmatrix} 0.03 \\ 0.02 \\ 0.01 \\ 0.00 \end{pmatrix} \tag{2}$$

by inspection.

Imposing the conservation of power so that the sum of the flows leaving a node is equal to the power injected at the node, P_i, leads to a system of linear equations:

$$S \cdot \theta = P, \tag{3}$$

where S is the matrix of transmission line susceptances,

$$S = \begin{cases} S_{ij} = \dfrac{-1}{x_{ij}}, \text{ i, j} \neq \text{reference node} & \text{(4a)} \\ S_{ii} = \displaystyle\sum_{j,\, j \neq i} \frac{1}{x_{ij}}, \text{ i} \neq \text{reference node} & \text{(4b)} \\ S_{ij} = 0, \text{ i, j = reference node} & \text{(4c)} \\ S_{ij} = 1, \text{ i = reference node} & \text{(4d)} \end{cases}$$

The column vectors θ and P represent the phase angles and power injections at each node, respectively. Note that the power-injection vector is normalized by an MVA base (typically 100 MVA), the power is positive when entering the grid (generation) and negative when leaving the grid (consumer load), the power injection at the reference node is always zero, and the phase angles are in radians.

Then our problem can be written numerically as

$$\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \theta = \begin{pmatrix} 0.01 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{5}$$

The inverse of the susceptance matrix $\underline{S}$ is

$$S^{-1} = \begin{pmatrix} 3 & 2 & 1 & 0 \\ 2 & 2 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

Multiplying right and left-hand sides of (5) by S^{-1} to solve for θ yields the θ expected.

Now let us examine the solution of this problem using diakoptics. Matching the variable names in Aitchison's algorithm, we have $d \equiv P$, $M \equiv S$, and $z \equiv \theta$.

First, partition the coefficient matrix. For our problem, let

$$A = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{7}$$

and

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{8}$$

Note that the equation $Ax = d$ can be solved without inverting A explicitly, but we will compute the explicit inverse for clarity.

$$A^{-1} = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{9}$$

Then x is

$$\begin{aligned} x &= A^{-1}d \\ &= \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0.01 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0.02 \\ 0.01 \\ 0 \\ 0 \end{pmatrix} \end{aligned} \tag{10}$$

Note that B has two nonzero columns ($r = 2$). Select these columns and then solve for the matrix X as the solution to the equation $X = A^{-1}B^r$. Note that X will have r columns and n rows, the same as B^r.

$$\begin{aligned} X &= A^{-1}B \\ &= \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \bullet \begin{pmatrix} 0 & 0 \\ 0 & -1 \\ -1 & 0 \\ 0 & 0 \end{pmatrix}. \\ &= \begin{pmatrix} 0 & -1 \\ 0 & -1 \\ -0.5 & 0 \\ 0 & 0 \end{pmatrix} \end{aligned} \tag{11}$$

Next, select the rows of X corresponding to the nonzero columns of B and solve for z_r.

$$\begin{aligned} (I + X_r)z_r &= x_r \\ \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & -1 \\ -0.5 & 0 \end{pmatrix}\right] z_r &= \begin{pmatrix} 0.01 \\ 0 \end{pmatrix}. \\ z_r &= \begin{pmatrix} 0.02 \\ 0.01 \end{pmatrix} \end{aligned} \tag{12}$$

Finally, calculate z_{n-r} and compose with z_r to find the final solution z.

$$\begin{aligned} z_{n-r} &= x_{n-r} - X_{n-r} \cdot z_r \\ &= \begin{pmatrix} 0.02 \\ NA \\ NA \\ 0 \end{pmatrix} - \begin{pmatrix} 0 & -1 \\ 0 & -1 \\ -0.5 & 0 \\ 0 & 0 \end{pmatrix} \bullet \begin{pmatrix} NA \\ 0.02 \\ 0.01 \\ NA \end{pmatrix}. \\ &= \begin{pmatrix} 0.03 \\ NA \\ NA \\ 0 \end{pmatrix} \end{aligned} \tag{13}$$

where the values absent from z_{n-r} have been represented explicitly by the value *NA* for clarity. Then the final solution is obtained by substituting the values from z_r for the *NA* values, obtaining

$$z = \begin{pmatrix} 0.03 \\ 0.02 \\ 0.01 \\ 0.00 \end{pmatrix} \tag{14}$$

as expected.

3.2 Near-Singularity of the Power-Flow Problem and Advantages of Diakoptics

The susceptance matrix, S, described by equations (4) is nearly singular. If not for the reference node, described by equations (4c,d), this susceptance matrix would be identically singular. Note also that the submatrix comprised of the rows and columns of S excluding those corresponding to the reference node is itself nearly singular, except for transmission lines adjoining the reference node whose susceptances are added to the diagonal elements corresponding to the neighboring nodes in the transmission system.

The near-singularity of the power-flow problem exacerbates the difficulty of the problem. In combination with the size of a typical problem, which may describe tens of thousands of nodes in a commercial transmission system, the solution of the power-flow problem can be computationally challenging. Of critical importance to the commercial electric-power industry, this problem has been studied extensively [10, 11, 12]. The near-singularity of the problem typically precludes the use of Krylov techniques for the solution.

Diakoptics partitioning mitigates the difficulty of the power-flow problem in two ways. First, obviously, the partitioning of Aitchison's matrix *A* reduces the size (number of nodes in each) of the individual independent problems. This partitioning speeds the solution of the individual components. Typically, depending on the solution method, the solution of these independent submatrices of *A* can be reused many times in Aitchison's step 2 (see example, equation (11)). Second, the partitioning reduces the proximity to singularity of these independent sub-problems. The partitioning described in equations (7,8) remove off-diagonal elements of the susceptance matrix (corresponding to the susceptances of transmission lines between partitions) without eliminating the contribution of these lines' susceptances to the diagonal elements (equation (4b)). So the solution of the sub-problems is improved by diakoptics, both by reducing the size of the coefficient submatrices to be solved and making these submatrices less singular.

4 Master/Slave Diakoptics Implementation

The diakoptics partitioning of the coefficient matrix into independent submatrix components suggests a parallel-computing method using a traditional master/slave implementation. Each submatrix is assigned to a slave computing node. Then Aitchison's algorithm steps 1 and 2 can be solved in parallel in straightforward fashion.

Aitchison's algorithm step 3 involves the solution of an $r \times r$ coefficient matrix, $(I + X_r)$, which may be non-sparse. In terms of a graph, the components of this matrix are assembled from the edges connecting the partitioned subgraphs residing on the computational slave nodes. In electrical-engineering jargon, these edges are called interties. Aggregating this coefficient matrix from the slaves' interties is a task that can be assigned to the computational master node. Then the completion of step 3 by the master node represents an algorithmic bottleneck, as the slave nodes wait for completion of this step by the master until Aitchison's algorithm step 4 can begin.

Optimal partitioning to reduce the order, *r*, of the $(I + X_r)$ coefficient matrix is important to optimizing the performance of the diakoptics algorithm. The order *r* is equal to the number of transmission-system nodes having interties incident. Note that this number is not necessarily equal to two times the number of interties, as multiple interties may be incident at the same node. Several techniques have been suggested for optimizing this partitioning [3, 6].

Aitchison's algorithm step 4 can be accomplished by passing the results of the calculation of z_r in step 3 to the slave nodes for computing step 4 in parallel. Although the computation of step 4 depends on a multiplication by X_{n-r} (which is an aggregate across all the slave nodes from the calculation performed in step 2), the only non-zero components of X_{n-r} contributing to the result of the multiplication that will be stored on a slave are those components that are stored locally on the slave. This is demonstrated in the example in equation (13).

This diakoptics algorithm for the calculation of power flow was tested empirically using Los Alamos National Laboratory's "Rockhopper" extreme-Linux supercomputer. The Rockhopper supercomputer uses a cluster of 128 dual-processor 500-MHz Pentium® III computers with MPI message passing using a Myrinet 1.28 Gbit/s communication infrastructure.

Several software methods were combined to solve the problems presented by the Aitchison algorithm. Solution of the submatrix problems on the slave computational nodes was accomplished using an optimized sparse symmetric matrix linear solver [8]. This linear solver supports the storage and re-use of coefficient-matrix factorization for improved solution speed in subsequent solutions. Because the matrix $(I + X_r)$ in Aitchison's step 3 may be both non-sparse and non-symmetric, a conventional LU-decomposition method was used for this step. This method also supported storage and re-use of the factorization, permitting subsequent solutions requiring only a backsubstitution procedure. Matrix partitioning was accomplished using a method based on the Karypis, et al. graph heuristic [6].

This implementation of the diakoptics algorithm for power-flow calculation exhibited expected scaling behavior. Figure 2 shows the performance of the diakoptics algorithm by plotting the time required to complete a power-flow simulation as a function of the number of slave processors. The power-flow simulation was a test of completing 1000 different power-flow calculations for the same transmission system. This study used a 9952-node, 12471-edge model of the transmission system of the Western Systems Coordinating Council (WSCC) obtained from the California Independent System Operator (Cal-ISO).

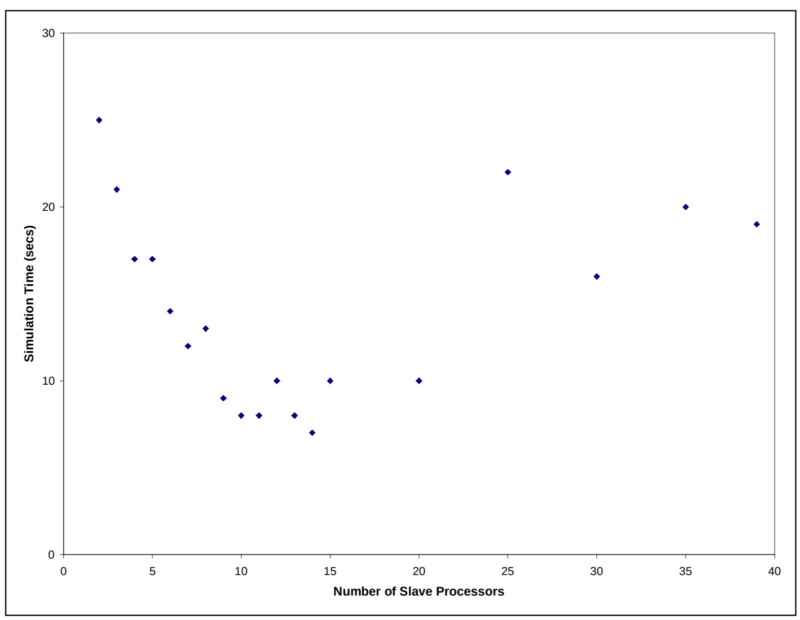
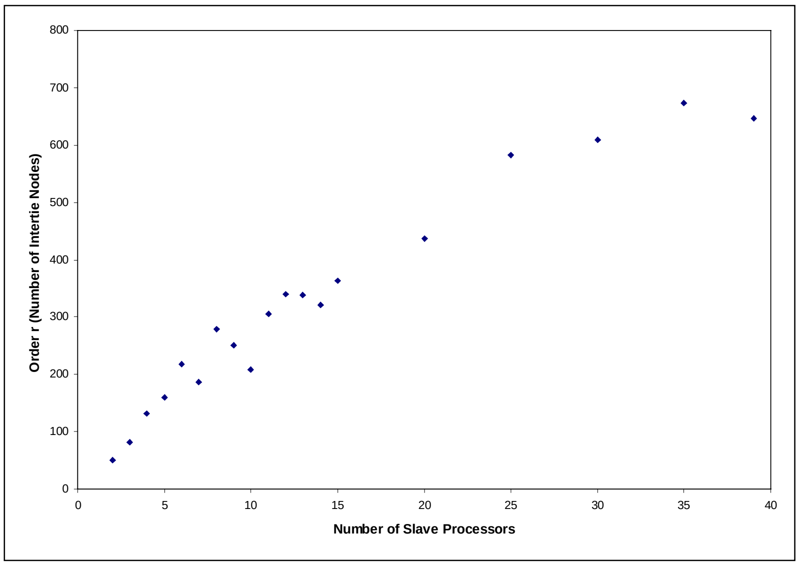


Figure 2. Simulation Time vs. Number of Slaves

Initially, the simulation time decreases quickly with the number of slave processor nodes. After reaching a minimum at 14 slave nodes, the simulation time then increases slowly as more slave nodes are used. The partitioning of the WSCC transmission system achieves partitions of approximately equal size, and the rank of each slave submatrix is approximately n/k where n is the total number of transmission-system nodes and k is the number of slave processors. However, as k increases, the number of intertie nodes required to partition the transmission system also increases. This number is the order, r, of the matrix $(I + X_r)$. As r increases, the time required for Aitchison's step 3 soon overwhelms the gain in performance achieved in steps 1 and 2 from increasing the number of slave nodes. Figure 3 shows r as a function of the number of slave processors.



Empirical analysis of these results indicates that the simulation time has an approximate O($r^2 + k^{-1}$) dependence on the number of intertie nodes, r, and the number of slave processors, k. These results model the dependence of the simulation time upon the performance of the Pissanetzky algorithm for small k and upon the results of the partitioning algorithm when r becomes large. Figure 4 shows the results of this empirical model of the simulation time.

6 Conclusions

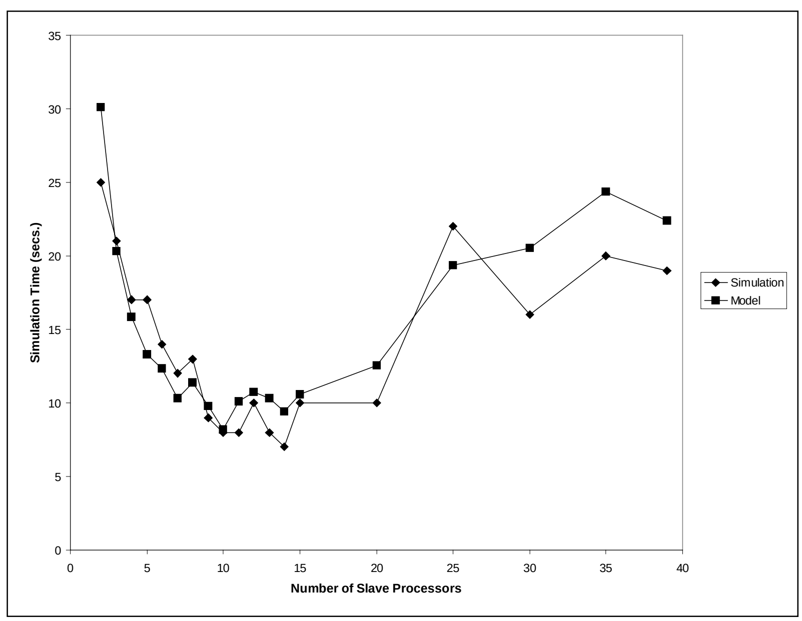
Diakoptics is a computational technique that presents performance advantages for solution of nearly-singular sets of sparse linear equations by a parallel computer with conventional master/slave architecture. Performance of the algorithm depends on a) having an efficient linear solver for the sparse equations on the slave nodes, b) having an efficient linear solver for the non-sparse equations on the master node, and c) efficient partitioning of the equations into independent sub-problems with minimal graphical interconnection. Solution of step 3 of Aitchison's diakoptics algorithm presents a performance bottleneck that depends strongly on the number of interconnections between the components of a block-diagonal coefficient matrix. Efficient partitioning to reduce the number of these interconnections is essential.

The diakoptics algorithm should be effective for computations of power flow in a commercial electric-power infrastructure. Sparsity and connectivity characteristics of commercial electric-power networks support the use of the diakoptics algorithm for computational performance improvement.

Additional work is indicated in several areas. First, exploration of metrics for quantifying the singularity of a matrix will strengthen the assertion that diakoptics reduces the singularity of the submatrices by changing the diagonal elements of otherwise-singular matrices. Second, scaling studies can be performed using synthetic networks, such as trees and nearly-regular graphs, to control the sizes of the partitioned networks and the interconnections between them to test the model for scaling performance. Third, further work is needed to test alternative distributed algorithms (e.g., PETSc [2]) for the Aitchison step 3, to evaluate possible performance improvements. Finally, the performance of alternative partitioning algorithms should be evaluated, as the dependence of the performance of the Aitchison algorithm on the number of intertie nodes presents a limit to the performance improvement that can be achieved through parallel processing.

Acknowledgments

This work was supported by the Electricity Industry Simulation System (ELISIMS) project at LANL. This project is funded by the LANL Associate Laboratory



Director for Strategic and Supporting Research. The effort is supported by the LANL Delphi Project, which uses the high-performance computing platforms of the LANL Accelerated Strategic Computing Initiative to address scientific problems of national importance.

The University of California operates LANL for the United States Department of Energy under contract *W-7405-ENG-36*. LANL strongly supports academic freedom and a researcher's right to publish; therefore, the Laboratory as an institution does not endorse the viewpoint of a publication or guarantee its technical correctness.

Intel® and Pentium® are registered trademarks of Intel Corp.

References

[1] Aitchison, P. W., *Diakoptics as a General Approach in Engineering*, J. Eng. Math. **21**, 47 (1987).

[2] Balay, S., et al., *PETSc 2.0 Users Manual*, Argonne National Laboratory technical report ANL-95/11, 1999.

[3] Chan, K. W., R. W. Dunn, and A. R. Daniels, *Efficient Heuristic Partitioning Algorithm for Parallel Processing of Large Power Systems Network Equations*, IEE Proc. Gener. Transm. Distrib. **142**, 625 (1995).

[4] Ipsen, C. F. and C. D. Meyer, *The Idea Behind Krylov Methods*, Am. Math. Monthly **105**, 889 (1998).

[5] Henderson, H. V. and S. R. Searle, *On Deriving the Inverse of a Sum of Matrices*, SIAM Review **23**, 53 (1981).

[6] Karypis, G., and V. Kumar, *A Coarse-Grain Parallel Formulation of Multilevel k-Way Graph Partitioning Algorithm*, Eighth SIAM Conference on Parallel Processing for Scientific Computing, 14-17 March 1997, Minneapolis, MN.

[7] Kron, G., *Diakoptics - Piecewise solution of Large Scale Systems*, Electrical Journal, in 20 parts from **158**, 1673 (1957) to **162**, 431 (1959).

[8] Pissanetzky, S., *Sparse Matrix Technology*, Academic Press, London, 1984.

[9] Press, W. H., et al., *Numerical Recipes in C*, Cambridge University Press, New York, 1990.

[10] Sato, N., and W. F. Tinney, *Techniques for Exploiting the Sparsity of the Network Admittance Matrix*, IEEE Trans. on Power Apparatus and Systems **82**, 944 (1963).

[11] Tylavsky, D. J., et al., *Parallel Processing in Power Systems Computation*, IEEE Trans. on Power Systems **7**, 629 (1992).

[12] Wang, F. Z., N. Hadjsaid, and J. C. Sabonnadière, *Power System Parallel Computation by a Transputer Network*, Electric Power Systems Research **52**, 1 (1999).

[13] Wood, A. J. and B. F. Wollenberg, *Power Generation, Operation, & Control*, J. Wiley and Sons, New York, 1984.

About the Authors

Dr. L. Jonathan Dowell is a Technical Staff Member in the Energy and Environmental Analysis Group at LANL. Dr. Dowell received his Ph.D. in engineering physics from the University of Virginia in 1989. Dr. Dowell has performed research and analysis in the fields of metrology, power-systems engineering, computer science, and applied mathematics. Dr. Dowell also is president of ReefNews, Inc., a non-profit organization that produces educational materials about the oceans.

Doug Roberts is a Technical Staff Member in the Technology and Safety Assessment Division at LANL. He has degrees in chemical and industrial engineering. For the past 25 years (the last 15 at LANL), Mr. Roberts has been involved with simulation-development activities which have resulted in a variety of object-oriented, C^3I, discrete-event, and distributed simulation designs.

Dr. Dale B. Henderson is a Technical Staff Member in the Technology and Safety Assessment Division at LANL. Dr. Henderson received his Ph.D. in applied physics from Cornell University in 1967. Dr. Henderson has performed basic, applied, and computational research in plasma physics, nuclear weapons physics, and large-scale simulation.